UNIVERSIDAD DE GUAYAQUIL

Facultad de Ciencias Matemáticas y Físicas

Carrera de Ingeniería en Sistemas Computacionales

"Cliente Seguro de I.M. con Jabber."

PROYECTO DE TESIS DE GRADO

CURSO DE GRADUACIÓN

Previo a la Obtención del Título de:

INGENIERO EN SISTEMAS COMPUTACIONALES

Autores:

Crespo Pérez Jenny Margarita
Figueroa Reyes Elsie Magaly
Vera Espinoza Ena María

Guayaquil - Ecuador

Año: 2007

INDICE GENERAL

Pág.

MANUAL TÉCNICO

1. Arquitectura del Sistema 2

2. Módulos de CSJabber 2

2.1 Módulo de inicio de sesión 3

2.2 Modulo de Chat 4

2.3 Modulo de agregar contacto 4

2.4 Modulo de Interfaz principal 5

2.5 Modulo de interacción. 6

2.6 Modulo SSL. 6

2.7 Modulo de Conexión. 7

2.8 Modulo de contactos 8

2.9 Modulo de mensajes. 9

2.10. Modulo de transferencia de archivo. 10

3. Paquetes utilizados para programar el programa cliente. 11

3.1 JSO (Java Stream Object) 11

3.2 Paquete org.jabberstudio.jso 14

3.3 Paquete org.apache.* 16

3.4 El paquete Java Preparación para Jabberd 2 17

4. Preparación para Jabberd 2 19

4.1Crear un usuario y agruparlo. 21

Pág.

4.2 Crear los directorios para PID y los registros. 21

4.3 Requisitos previos a la instalación. 22

5. Instalación del Servidor 24

6. Configuración Básica del Jabberd 26

6.1 Fijar el Hostname 27

6.2 Configurar la autentificación y paquetes de almacenaje 28

6.3 Probar el servidor 31

7. Crear usuarios desde la consola de MySQL 33

8. Generar un SSL (Certificado) en el servidor Jabberd 2 34

8.1 Asignar un certificado para uso de los clientes de Jabber 35

8.2 Comunicación de Non-STARTTLS inhabilita 36

8.3 Asignar un certificado para uso de los componentes de Jabberd 36

9. Descripción de la arquitectura de Jabberd 37

9.1 Cliente Jabber a Servidor Jabber 39

9.2 Servidor Jabber a Servidor Jabber 39

9.3 Servidor Jabber a otra IM 40

10. Instalación del Cliente 41

10.1 Instalación bajo Sistema Operativo Linux 41

10.2 Instalación bajo Sistema Operativo Windows XP 42

11. Estándar de Desarrollo del Proyecto 42

11.1 Identificador de tipo de variables. 43

Pág.

11.2 Clase de String y Wrappers(envolturas) 44

11.3 Nombres para procedimientos 44

11.4 Nombres para funciones 45

11.5 Nombres de las clases 45

11.6 Componentes visuales (Swing) 46

12. Descripción de la Bases de Datos 46

13. Diccionario de la Base de Datos 48

MANUAL DE USUARIO

1 Propósito 58

2 Aplicabilidad 58

3 Principio de Interfaz de Usuario 59

3.1 Elementos de la Ventana Principal 59

3.2. Menú Archivo 60

3.2.1. Inicio Sesión 60

3.2.2. Cerrar Sesión 65

3.2.4. Transferir Archivo 65

3.2.5. Estado 67

3.2.6. Cerrar 68

3.3. Menú Contactos 68

3.3.1. Agregar Contacto 68

Pág.

3.3.2. Eliminar Contacto 69

3.4. Ayuda 69

3.4.1. Contenido 70

3.4.2. Acerca del Sistema 70

Recomendaciones 71

ÍNDICE DE GRÁFICOS

		Pág.
Gráfico 1	JSO (Java Stream Object)	11
Gráfico 2	Simbología de la arquitectura de Jabber	38
Gráfico 3	Arquitectura de Jabber	38
Gráfico 4	Descripción de la base de datos	47

ÍNDICE DE TABLAS

		Pág.
Tabla 1	Core API	12
Tabla 2	Core XMPP Stream Features	13
Tabla 3	Jabber/XMPP Extensions	13
Tabla 4	Resumen de la Interfaces	14
Tabla 5	Resumen de Clases	15
Tabla 6	Resumen de excepciones	15
Tabla 7	Paquete Apache	16
Tabla 8	Clases utilizadas	16
Tabla 9	Java 2 paquetes de la plataforma	17
Tabla 10	Preparación para montar servidor Jabber2	19
Tabla 11	Requisito de Instalación	22

MANUAL TÉCNICO

1. **Arquitectura del Sistema**

El sistema CSJabber posee una arquitectura ***CLIENTE-SERVIDOR***. Bajo este esquema, la comunicación de una máquina cliente que hace conexión a un servidor Ejabberd, a través de un puerto de comunicación específico, cuyo cliente puede ser ejecutado en diferentes sistemas operativo, pero el servidor deber ser ejecutado en arquitectura Linux.

El funcionamiento de nuestro aplicación es fácilmente comprensible se realiza de la siguiente manera el cliente ingresa el correo electrónico y clave correspondiente que son enviadas al servidor que verifica si el usuario y clave son correctos. Si el usuario ingresado es correcto el servidor devolverá todos los contactos que dicho usuario posee y además el estado de sus contactos actualmente.

El sistema además permite envío de mensajes entre los contactos que estén registrados para los usuarios del sistema, enviar avatares para mejorar la interacción de los usuarios.

2. Módulos de CSJabber

El programa consta de una serie de módulos que guardan estrecha relación, para lograr la funcionalidad del Cliente Seguro de I.M. con Jabber. Hemos clasificados los módulos según la función que realizan en modulo de inicio,

de sesión, Chat, agregar contacto, mensajes, interacción, SSL, interfaz principal, conexión, contactos, y transferencia de archivo.

2.1 Módulo de inicio de sesión

Nombre interno Login. Este modulo es una ventana que contendrá las opciones para que el usuario inicie sesión, consta de los campos usuario, password, servidor, los botones de conectar y salir y el checkbox para usar SSL.

Aquí presentamos un extracto de su código, específicamente la clase principal.

```
class Login extends JFrame implements ActionListener {
    JPasswordField jpf_psr;
    JTextField txt_usr;
    JTextField txt_host;
    JButton btn_aceptar;
     JButton btn_salir;
    JCheckBox ssl;
    JLabel lbl_usr;
    JLabel lbl_psr;
    JLabel lbl_host;
  //Constructor,
  public Login(String titulo, MainGUI mg) {
    super(titulo);
    Imprimir("uni@im.net", "", new String("im.net"), mg);
  }
```

Se debe hacer uso del modulo conexión.

2.2 Modulo de Chat

El modulo de Chat nos permite crear el formulario para el inicio de conversación entre dos clientes CSJabber, consta de una lista, una caja de texto un botón de envío de mensaje y un conjunto de botones que contienen smiley que pueden ser enviados en el momento de la conversación.

Los smiley se encuentran en formato .gif en una carpeta dentro del cliente llamada imágenes.

A continuación se muestra el ActionPerformed de un botón con smiley.

```
private void ic8ActionPerformed(java.awt.event.ActionEvent evt) {
    txtmensaje.setText(txtmensaje.getText()+":R");
  }
```

2.3 Modulo de agregar contacto

Nombre interno nuevocontacto. Este modulo crea una ventana que permite el ingreso del jid del usuario a insertar como nuevo contacto, con los JLabel “Correo” y dos botones uno para Agregar y otro para cancelar.

El botón agregar se encargará de agregar el contacto, y el botón cancelar oculta la ventana y no se realiza ninguna acción.

```
public void actionPerformed (ActionEvent e)
  {
     hide();
  }
```

El siguiente codigo nos permite obtener el jid del usuario a ingresar:

```
public String getJid() {
     return jid.getText();
  }
```

2.4 Modulo de interfaz principal

Contiene la interface principal del programa consta de una ventana en que se muestra los menús: Archivo, Contacto y Ayuda, muestra también la lista de contactos conectados

En este modulo se inicia el modulo Login, se inicia la conexión con el servidor Jabber. Se envía el estado de presencia inicial de los usuarios, se muestra la lista de contactos a través del modulo arbolcontactos, actualiza la lista de contactos, desconecta al usuario del servidor, inicializa y finaliza el hilo de conexión, cierra el programa, añade los contactos nuevos, envía las suscripciones a ese contacto, cambia el estado de los contactos en el árbol.

2.5 Modulo de interacción.

El modulo interactúa se encarga de gestionar diferentes eventos importantes en nuestro CSJabber Cliente para que funcione de manera correcta.

Los eventos sobre los que actúa son los mensajes, estados y las suscripciones, este modulo además recibe la conexión, lista de contactos

2.6 Modulo SSL.

Nombre interno modulSSL. Este modulo se encarga de enviar los datos del usuario que se va a conectar al servidor por medio de una conexión SSL.

Como los servidores Jabber utilizan certificados firmados por ellos mismos, para que podamos reconocer los certificados del servidor se debió crear una clase que es llamada, para que acepte siempre los certificados del servidor y no vote ninguna excepción.

```
class TrustEverythingManager implements X509TrustManager {
    public TrustEverythingManager() {
    }
        public X509Certificate[] getAcceptedIssuers() {
        return new X509Certificate[0];
    }
        public void checkClientTrusted(X509Certificate[] certs, String
authType) throws CertificateException {
    }
        public void checkServerTrusted(X509Certificate[] certs, String
authType) throws CertificateException {
    }
}
```

Para que el cliente pueda iniciar una conexión SSL debe seleccionar el Jcheckbox de SSL.

2.7 Modulo de conexión.

Este módulo permite la conexión entre el cliente y un servidor Jabber y todas las funcionalidades del cliente.

Una de las validaciones más importantes es la conexión con el servidor ya sea en el puerto 5222 o 5223 y también se evalúa el puerto 7777 que se utiliza para la transferencia de archivo.

```
if(port ==7777)
        conex.connect(sok_TLS                    =                    new
StartTLSSocketStreamSource(servidor.getDomain(), port));
           else
                if (port==5222)
        conex.connect(sok_TLS = new
StartTLSSocketStreamSource(servidor.getDomain(), port));
          else if (port==5223) {
        try {
              System.out.println("SSL");
           conex.connect(sok_TLS = ModulSSL.crearSSL(servidor, port));
          }
          catch (CouldNotConfigureSSLException e) {}
           catch (IOException e) {}
          }
```

En este modulo también se realiza la autentificación del cliente y nos permite configurar para que el cliente escuche el servidor, constantemente en la recepción de mensajes.

2.8 Modulo de contactos.

Este modulo permite recuperar nuestros contactos a partir de la libreta de direcciones que se encuentra almacenada en el servidor, en el momento en que nos conectamos. Nos permite conocer el estado de mis contactos en el momento de la conexión y así mismo que ellos conozcan el nuestro

Los principales atributos de esta clase son:

```
private int numero;
private Vector vec_contacto ;
private String str_subscription_status[];
private String str_estado[];
```

La petición de los contactos se realiza en

```
paquete = (InfoQuery)PacketMonitor.sendAndWatch(conn, paquete);
```

En caso de no existir la lista esta se creará vacía. Esta clase se relaciona con interactua.java para actualizar la lista de contactos.

2.9 Modulo de mensajes.

Este modulo permite enviar y recibir mensajes con emoticones o smiley. Nos permite enviar un mensaje encriptado o no.

Para la creación de un mensaje se implementa el siguiente código, primero se valida si va encriptado o no.

```
if (boo_encriptar)
                    {    Message _salida =
(Message)sdf.createPacketNode(sdf.createNSI("message",
conn.getDefaultNamespace()),Message.class);
                              _salida.reset();
                              _salida.setType(Message.CHAT);
                              _salida.setTo(receptor);
                              _salida.setFrom(_conexion.ftn_getJID() );
                              _salida.setBody(mensaje);

    _conexion.ftn_setStartTime(System.currentTimeMillis());
        try ……
```

Para recibir un mensaje se hace uso del siguiente código, pero primero se lo envía a la función para desencriptar en caso de que lo este.

```
public String ftn_recibir_mensaje (conexion _conexion , Message mensaje)
        {
try { Message desmensaje=_claves.ftn_desxifrarMissatge(mensaje);
                mensaje = desmensaje;
        }
```

2.10 Modulo de transferencia de archivo.

Su nombre interno es transarchivo.java este nos provee de funcionalidad para realizar la transferencia de un archivo entre clientes Jabber. Para ello se hace uso del puerto 7777.

Para enviar archivo debemos cumplir requisitos como capturar el nombre del archivo y el directorio donde se encuentra ubicado.

```
private void BuscarActionPerformed(java.awt.event.ActionEvent evt) {
        String dir =Preferences.getString("filetransfer", "uploadDir",null);
        fi =Buscar.getSelectedFile();
        Preferences.putString("filetransfer", "uploadDir",fi.getParent());
                System.out.println(Buscar.getSelectedFile());
            }
```

3. Paquetes utilizados para programar el programa cliente.

3.1 JSO (Java Stream Object)

Para que pueda interactuar el programa hecho en java con el servidor Jabber hay que hacer uso del paquete JSO, que debe incluirse dentro de los ejecutables para que corra el programa cliente.

JSO puede ser descargado del sitio Web www.jabberstudio.org, así como la descripción total del paquete.

Por medio del Forte For Java se observa la siguiente estructura, en que se agrupan las clases.

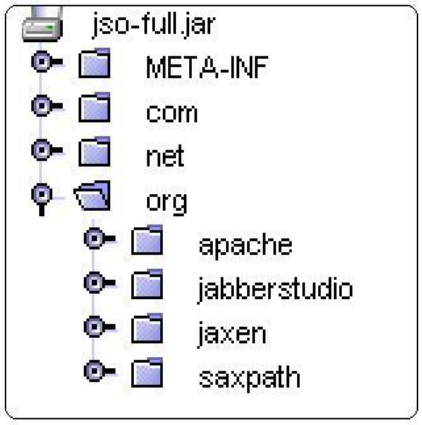


Gráfico 1

JSO (Java Stream Object)

Fuente: Programadores

Hacemos referencias al API (Application Programming Interface de Java). La referencia del API describe los interfaces, las clases, los métodos, y los campos individuales en JSO.

La versión de JSO es la 0.12.5 el autor es linuxwolf.

A continuación se describe solo los API utilizados en la programación del CSJabber, ya que JSO de muchas clases más.

Core API	
org.jabberstudio.jso	Define la interface básica "core" y clases para Jabber™ Stream Objects (JSO).
org.jabberstudio.jso.event	Provee clases e interfaces para los eventos generados por JSO.
org.jabberstudio.jso. features	Provee clases e interfaces para consumir StreamFeatures.
org.jabberstudio.jso.util	Proporciona clases de utilidad througout usadas para uso general del paquete JSO.
org.jabberstudio.jso.xpath	Proporciona los interfaces y las utilidades específicos al proceso de XPath de JSO. Características de la corriente de la base XMPP.

Tabla: 1

Fuente: org.jabberstudio.org

Core XMPP Stream Features	
org.jabberstudio.jso.sasl	Define las interfaces y clases para la ayuda o soporte SASL.
org.jabberstudio.jso.tls	Define los interfaces y las clases para la ayuda de STARTTLS.

Tabla 2

Fuente: org.jabberstudio.org

Jabber/XMPP Extensions	
org.jabberstudio.jso.x.core	Define interfaces Extension que son parte del bosquejo XMPP.
org.jabberstudio.jso.x.disco	Define interfaces Extensión y StreamElement para el descubrimiento del servicio, según lo documentado en JEP-0030.
org.jabberstudio.jso.x.info	Define un número “informational” interfaces Extension interfaces, según lo documentado por Fundación del software de Jabber.

Tabla 3

Fuente: org.jabberstudio.org

3.2 Paquete org.jabberstudio.jso

A continuación describiremos solo las clases e interfaces utilizadas en CSJabber, puesto que son muchas más de las especificadas en este manual.

Resumen de la Interfaces	
Packet	Interfaces primaria para Jabber y el stream XML
Presence	Encapsula un elemento de la presencia. (presence).
Stream	Interface Común que define todos los tipos de conexión.
StreamContext	Encapsula la información detallada para una "dirección" específicada en el stream.
StreamDataFactory	Interface para crear los datos del stream.
StreamFeatureset	Interface que representa el set de instancia StreamFeature.

Tabla: 4

Fuente: org.jabberstudio.org

Resumen de Clases	
JID	Representación de un objeto para identificar el JID dentro de Jabber.
JSOImplementation	Punto de comienzo para la librería JSO

Tabla 5

Fuente: org.jabberstudio.org

Resumen de excepciones	
StreamException	Root para todos los errores serios, Stream-related dentro de JSO.

Tabla 6

Fuente: org.jabberstudio.org

3.3 Paquete org.apache.*

Solo se describirán las clases que se utilizan dentro de programación del cliente seguro de mensajería instantánea para Jabber.

Paquete	
org.apache.log4j	Es el paquete principal log4j

Tabla 7
Fuente: www.apache.org

Las clases utilizadas son:	
java.lang.Object | +--org.apache.log4j.Category | +--org.apache.log4j.**Logger**	Logger es la clase central del paquete log4j. La mayoría de las funciones crean a través de esta clase, excepto la configuración.
java.lang.Object | +--org.apache.log4j.Priority | +--**org.apache.log4j.Level**	Define el set mínimo de niveles reconocido por el sistema esto es OFF, FATAL, ERROR, WARN, INFO, DEBUG, TRACE, y ALL. La clase LEVEL puede ser la subclase para definir un set de niveles más grande.

Tabla 8

Fuente: www.apache.org

3.4 El paquete Java

Se utiliza la plataforma de Java para la programación los principales paquetes de Java implementados son:

Java 2 paquetes de la plataforma	
java.awt	Contiene todas las clases para crear interfaces, gráficos y las imágenes.
java.awt.color	Proporciona las clases para los espacios de color.
java.awt.event	Proporciona interfaces y las clases para ocuparse de diversos tipos de acontecimientos de los componentes de AWT.
java.io	Proporciona al sistema entrada y salida a través de secuencias de datos, de la serialización y del sistema de ficheros.
java.lang	Proporciona las clases que son fundamentales al diseño del lenguaje de programación de Java.
java.math	Proporciona las clases para realizar la aritmética arbitrary-precision del número entero (BigInteger) y la aritmética decimal arbitrary-precision (BigDecimal).

java.net	Proporciona las clases para el establecimiento de una red en ejecución.
java.security	Proporciona las clases y los interfaces para el marco de la seguridad.
java.util	Contiene el marco de las colecciones, las clases de la colección de la herencia, el modelo del acontecimiento, las instalaciones de fecha y de la hora, la internacionalización, y las clases para uso general misceláneas (un tokenizer de la secuencia, un generador de número aleatorio).
javax.crypto	Proporciona las clases y las interfaces para las operaciones criptográficas.
javax.net	Proporciona las clases para los usos del establecimiento de una red.
javax.net .ssl	Proporciona las clases para el paquete ssl del socket.
Java.swing	Proporciona un sistema, para trabajar igual en todas las plataformas.

Tabla 9

Fuente: www.javasun.com

4. Preparación para Jabberd 2

Para montar un servidor Jabberd2 dentro de una intranet se especifican las siguientes acciones detalladas en la siguiente tabla a manera de resumen:

Parámetro	Requerido	Descripción	Tu información
Usuario y grupo de Jabberd	si	El usuario y el grupo del Linux (o el otro OS) que serán utilizados para funcionar Jabberd	*usuario: grupo del* jabber*:* jabber
Directorio de PID	Si	Directorio en el cual Jabberd almacena archivos de PID	/usr/local/var/jabberd/pid
Directorio del registro	No	Directorio para los registros de Jabberd. Si no especificado en archivos de la configuración, defectos de registración al syslog.	/usr/local/var/jabberd/log

Paquete de la autentificación	si	Paquete que se utilizará para la gerencia de la autentificación de Jabberd	DB de MySQL, de PostgreSQL, de Berkeley, OpenLDAP o PAM
Paquete del almacenaje de datos	Si	Paquete para el almacenaje de los datos de Jabberd	DB de MySQL, de PostgreSQL o de Berkeley
Usuario y contraseña de MySQL	MySQL	Usuario y contraseña de MySQL que Jabberd utiliza al conectar con MySQL	*usuario: contraseña* jabberd2*:* secreto
Localización dominante del SSL	N	Localización de OpenSSL pemfile. Requerido para la comunicación SSL-cifrada	/usr/local/etc/jabberd/server.pem

Tabla 10

Fuente: www.jabberes.org

4.1 Crear un usuario y agruparlo.

Se debe crea un usuario y un grupo jabber para hacer funcionar el servidor. Escribir lo siguiente:

```
groupadd jabber
useradd -g jabber jabber
```

4.2 Crear los directorios para PID y los registros.

Debes crear un directorio para que Jabberd almacene su PID y ficheros de diario, y la propiedad de estos directorios se debe de fijar de antemano por el usuario.

```
mkdir /usr/local/var
mkdir /usr/local/var/jabberd
mkdir /usr/local/var/jabberd/pid
chown -R jabber.jabber /usr/local/var
ls -ld /usr/local/var
drwxr-sr-x  3 jabber jabber 4096 2007-02-26 10:12 /usr/local/var
```

Directorio log (opcional): Si deseas crear un directorio separado para los registros de Jabberd, y fijar la propiedad a tu usuario de Jabber:

```
mkdir /usr/local/var/jabberd/log
 ls -lh /usr/local/var
total 4,0K
drwxr-sr-x  4 jabber jabber 4,0K 2007-02-26 10:14 jabberd
ls -lh /usr/local/var/jabberd/
total 8,0K
drwxr-sr-x  2 root   jabber 4,0K 2007-02-26 10:14 log
drwxr-sr-x  2 jabber jabber 4,0K 2007-02-26 10:12 pid
```

4.3 Requisitos previos a la instalación.

Jabberd 2 tiene cuatro requisitos previos:

- OpenSSL (versión 0.9.6b o superior)
- Libidn (versión 0.3.0 o superior)
- Paquete del almacenaje de datos
- Paquete de la autentificación

OpenSSL	OpenSSL proporciona al cliente cifrado hacia el servidor. El protocolo de XMPP requiere que la ayuda TLS (capa de los servidores de Jabber de la seguridad del transporte). TLS es el sucesor al SSL.

Libido	Libidn proporciona la funcionalidad necesaria de la manipulación de secuencia para Jabberd 2. Antes de jabberd 3, el libidn fue incluido con la distribución de Jabberd 2; sin embargo, un conflicto de licencia hace necesario que el libidn esté instalado por separado.
Paquete de almacenaje de datos	Jabberd 2 puede utilizar una de tres bases de datos libres para proporcionar almacenaje de datos: *MySQL* * *DB de Berkeley (4.1.24 o más altos)* * *PostgreSQL* MySQL es el almacén recomendado y por defecto de los datos.
Paquete de autenticación	Jabberd 2 puede utilizar los siguiente paquete para manejar la autentificación del usuario: *MySQL*, *DB de Berkeley* (versión 4.1.24 o superior, *PostgreSQL*, *OpenLDAP* (versión 2.1.0 o superior), *PAM*

Tabla 11

Fuente: org.jabberstudio.org

Para el almacén de la base de datos se va a trabajar con MySQL por poseer en sus paquetes soporte de autenticación de usuarios y porque es la base de datos más recomendada por los creadores de Jabberd 2.

Nota: Antes de instalar el servidor debe estar instalada la base de datos MySQL

5. Instalación del Servidor

Se deben realizar los siguientes pasos:

1. **Descargar,** el **archivo jabberd-2.0sn.tar.gz** de la página Web de estudio Jabber, donde está la versión “n” representa la última versión de Jabberd 2.

   Se recomienda que sea en /home para construir los archivos de instalación

2. **Extraer** el archivo descargado con el siguiente comando:

   ```
   tar -zxvf  jabberd-2.0s3.tar.gz
   ```

3. **Configure la variable de ambiente**:

   ```
   export LDFLAGS =”-L/usr/local/lib –L/usr/lib/mysql”
   ```

4. **Configure el directorio local**, la base de datos MySQL, ssl para las seguridades, y el debug para los errores.

   ```
   ./configure --prefix=/usr/local --enable-idn --enable-mysql --enable-ssl --enable-debug
   ```

5. **Construir Jabberd**

```
make
chmod -R 640 /usr/local/etc/jabberd/*
ln -s /usr/local/etc/jabberd/ /etc/jabberd
```

6. **Instalar Jabberd**

```
make install
```

7. **Localización de archivos por defecto**

```
/usr/local/etc/jabberd    //Jabberd Configuration Files
/usr/local/bin //Jabberd Binaries(jabberd, c2s, resolver, router, s2s, sm)
```

8. **Fijar propiedades a los archivos de configuración,** los archivos de configuración de Jabberd contienen contraseñas; por lo tanto, se debe fijar propiedad y permisos en estos archivos de modo que sean solamente legibles por nuestro usuario del Jabber y escribibles por el root solamente. Con la localización de tus archivos de configuración y de tu usuario de Jabber, fijar la propiedad de estos archivos:

```
chmod -R 640 /usr/local/etc/jabberd/*
```

9. *Opcional***: Crear Symlink para los archivos de la configuración,** si utilizaste las localizaciones de archivo por defecto al instalar Jabberd, puedes desear crear un symlink (como superusuario) en /etc para los

archivos de la configuración. Esto hará que sea más fácil encontrarlos y corregirlos:

```
Ln -s /usr/local/etc/jabberd/ /etc/jabberd
```

Hasta aquí Jabberd 2 ya esta instalado, lo siguiente son las configuraciones.

6. Configuración Básica del Jabberd

Jabberd 2 se configura vía sus seis archivos de XML. Para las instalaciones del defecto, estos archivos de la configuración se pueden encontrar en /usr/local/etc/jabberd/, y son accesibles de /etc/jabberd si creaste el symlink para este directorio. Hay que seguir estos tres pasos:

- Fijar el hostname (sm.xml y c2s.xml).
- Configurar la autenticación y paquetes de almacenaje
- Probar el servidor

6.1 Fijar el Hostname

Modificar

/etc/jabberd/c2s.xml línea 63

//editara el archivo configurar el nombre, IP, puerto

c2s.xml :

```
  <!-- Local network configuration -->
  <local>
  <!-- Who we identify ourselves as. This should correspond to the
       ID (host) that the session manager thinks it is. You can
       specify more than one to support virtual hosts, as long as you
       have additional session manager instances on the network to
       handle those hosts. The realm attribute specifies the auth/reg
       or SASL authentication realm for the host. If the attribute is
       not specified, the realm will be selected by the SASL
       mechanism, or will be the same as the ID itself. Be aware that
       users are assigned to a realm, not a host, so two hosts in the
       same realm will have the same users.
       If no realm is specified, it will be set to be the same as the
    ID. -->

   <id>somemachine.somedomain.com</id>
```

Nombre de la maquina <id> servidor 10.10.10.4 con la IP o el nombre del servidor.

/etc/jabberd/sm.xml línea 1

```
sm.xml :
 <!-- Session manager configuration -->
   <sm>
 <!-- Our ID on the network. Users will have this as the domain part of
       their JID. If you want your server to be accessible from other
       Jabber servers, this ID must be resolvable by DNS.s
       (default: localhost) -->
 <id>somemachine.somedomain.com</id>
```

6.2 Configurar la autentificación y paquetes de almacenaje

Configurar la base de datos. Jabberd 2 requiere la versión 4.0 de MySQL

//archivo /var/lib/mysql/jabberd2/ db-setup.mysql

Ahora que existe una base de datos para Jabberd en el directorio de datos de MySQL, hay que crear un usuario de MySQL que el servidor pueda utilizar para conectarse con MySQL.

De la consola de MySQL, se debe ejecutar la declaración del SQL (abajo), substituyendo “secret” por la contraseña que usted ha elegido para su usuario de Jabberd MySQL:

```
Crear usuario, setear el password y darle los permisos sobre la base

GRANT select,insert,delete,update ON jabberd2.*

to jabberd2@localhost IDENTIFIED by 'secret';
```

“secret” es la contraseña usada por defecto para Jabberd.

Se debe crear un enlace simbólico si no existe poner:

```
    ln -s /var/lib/mysql/mysql.sock /tmp/mysql.sock     //enlace simbólicos
```

Ahora se debe configurar para el almacenaje usando MySQL (sm.xml) modificar sm.xml para que Jabberd2 haga uso de MySQL.

```
<!-- Storage database configuration -->
 <storage>
  <!-- By default, we use the MySQL driver for all storage -->
     <driver>mysql</driver>
```

En sm.xml bajo la sección etiquetada configuración del conductor de MySQL, sustituir el secreto por su contraseña de MySQL. Cambiar a usuario si no está utilizando el usuario por defecto que es “*jabberd2*”:

```
<!-- MySQL driver configuration -->
      <mysql>
        <!-- Database server host and port -->
        <host>localhost</host>
        <port>3306</port>
```

```
    <!-- Database name -->
    <dbname>jabberd2</dbname>

    <!-- Database username and password -->
    <user>jabberd2</user>
    <pass>secret</pass>

    <!-- Transaction support. If this is commented out, transactions
         will be disabled. This might make database accesses faster,
         but data may be lost if jabberd crashes.

         This will need to be disabled if you are using a MySQL
         earlier than v3.23.xx, as transaction support did not appear
         until this version. -->
    <transactions/>
  </mysql>
```

Configurar para la autentificación usando MySQL (c2s.xml).

A continuación se va usar MySQL para la autentificación. La configuración de la autentificación de Jabberd 2 para MySQL es igual que arriba, salvo que la información se encuentra en c2s.xml.

Paquete de datos de autentificación. En c2s.xml bajo sección etiquetada la configuración de la base de datos de la autentificación/del registro, cerciórese de que el conductor es mysql. Se debe fijar por defecto:

```
<!-- Authentication/registration database configuration -->
    <authreg>
   <!-- Backend module to use -->
      <module>mysql</module>
```

En c2s.xml bajo la etiquetada configuración del módulo de MySQL, substituir “secret” por su contraseña de MySQL. Cambiar el usuario si no estás utilizando a usuario del defecto (jabberd2):

```
<!-- MySQL module configuration -->
     <mysql>
       <!-- Database server host and port -->
       <host>localhost</host>
       <port>3306</port>

       <!-- Database name -->
       <dbname>jabberd2</dbname>

       <!-- Database username and password -->
       <user>jabberd2</user>
       <pass>secret</pass>
     </mysql>
```

El almacenamiento y la autenticación está completa, ahora testear el Servidor. Se debe cambiar el anfitrión solamente si tu servidor de MySQL está funcionando en un anfitrión distinto. Debes cambiar el puerto solamente si tu servidor de MySQL está funcionando en un puerto no estándar (el puerto 3306 es el defecto para las instalaciones de MySQL).

6.3 Probar el servidor

Antes de realizar más configuraciones, se debe probar el servidor. Debes poder encender y probar tu servidor de Jabberd 2 usando la escritura del

arranque de Jabberd 2 (como el usuario del jabber). No es recomendable utilizar a root para correr el demonio de jabberd2 sino por medio de otro usuario:

```
su –l jabber                ./jabberd        //correr jabber
```

Si Jabberd no comienza, cerciorarte de que cualquier caso anterior haya parado. Estos casos incluyen todos los componentes runtime de Jabberd (jabberd, router, resolver, sm, s2s and c2s). Observa que tu usuario del jabber no puede tener path por default; por lo tanto, debes hacer **cd** al directorio jabberd /usr/local/bin y luego correr *jabberd*.

Comprobar que tus servidores elegidos del paquete de los datos estén funcionando (excepto el DB de Berkeley, que no requiere comenzar). Comprobar tu syslog para saber si hay mensajes de error. Si tu servidor no puede comenzar, puedes comenzar Jabberd 2 con la opción de eliminar errores. Tipear lo siguiente:

```
/usr/local/bin/jabberd –D
```

7. Crear usuarios desde la consola de MySQL

Editar sm.xml para permitir autocrear a los usuarios

Ahora que auto-crear esta habilitado, puedes crear usuarios desde la consola de MySQL.

Logoneate dentro de la consola de MySQL como usuario Jabberd2:

```
mysql -u jabberd2 –p
```

Desde la consola de MySQL, enciende la base de datos jabberd2:

```
mysql>use jabberd2
```

Desde la consola de MySQL, inserta dentro de la tabla "**authreg**". La fila debe contener valores para **username**, **realm** y **password:**

```
mysql>insert into authreg (username, realm, password)
        -> values ('myusername', 'somedomain.com', 'mypassword');
```

Cambia los valores de somedomain.com.

8. Generar un SSL (Certificado) en el servidor Jabberd 2

Con el siguiente comando se empieza a generar una llave de confianza.

```
openssl req -new -x509 -newkey rsa:1024 -days 3650 -keyout privkey.pem –out server.pem
```

Se crea una llave para 3650 días o sea diez años de vida.

Con el siguiente comando removemos el passphrase de nuestra llave privada:

```
openssl rsa -in privkey.pem -out privkey.pem
```

Ingrese este comando para combinar la clave privada y pública en un solo archivo:

```
cat privkey.pem >> server.pem
```

Anule la llave privada: rm privkey.pem

Ahora usted podrá mover la llave a un lugar permanente. Por ejemplo, para mover la llave al directorio de jabberd predefinido, usted pondrá esta orden como súper usuario:

```
mv server.pem /usr/local/etc/jabberd/
```

Ahora se debe poner los permisos en este archivo para que sean del súper usuario:

```
chown root.jabber /usr/local/etc/jabberd/server.pem //certificado del servidor server.pem
chmod 640 /usr/local/etc/jabberd/server.pem
```

Ahora el certificado esta listo para usarse.

8.1 Asignar un certificado para uso de los clientes de Jabber

La llave del SSL para los clientes de Jabber está situada en c2s.xml. Observar que c2s.xml contiene la localización de la llave del SSL usada por los clientes de Jabber, además de la localización de la llave del SSL usada para c2s comunicaciones router.

Localización de la llave SSL: Descomentar “pemfile” (de tu llave del SSL) bajo la sección etiquetada configuración de red local, y edite para la localización de tu llave SSL.

Observar que si tu archivo del PEM está en la localización del default /usr/local/etc/jabberd/server.pem, necesitas solamente descomentar está sección como abajo:

```
<!-- File containing a SSL certificate and private key for client
     connections. If this is commented out, clients will not be
     offered the STARTTLS stream extension -->
 <pemfile>/usr/local/etc/jabberd/server.pem</pemfile>
```

Si deseas poner el SSL en el puerto 5223, descomentar el SSL-puerto etiqueta:

```
<ssl-port>5223</ssl-port>
```

8.2 Comunicación de Non-STARTTLS inhabilita

Para requerir comunicaciones de STARTTLS, descomentar la etiqueta del require-starttls como abajo:

```
<!-- Require STARTTLS. If this is enabled, clients must do STARTTLS
     before they can authenticate. Until the stream is encrypted,
     all packets will be dropped. -->
<require-starttls/>
```

8.3 Asignar un certificado para uso de los componentes de Jabberd

Cada uno de los cinco componentes de Jabberd tiene su propia configuración para las comunicaciones cifradas component-to-router. Así,

estos archivos de configuraciones se deben corregir para proporcionar la comunicación segura entre los componentes de Jabberd:

- router.xml
- sm.xml
- resolver.xml
- s2s.xml
- c2s.xml

9. Descripción de la arquitectura de Jabberd

Este es un bosquejo de cómo funciona Jabber. Bajo el protocolo XMPP, proporciona un protocolo para la mensajería, y proporciona una plataforma estandardizada para la comunicación del servidor de Jabber:

- Cliente de Jabber al servidor de Jabber
- Servidor de Jabber al servidor de Jabber
- Servidor de Jabber a otra IM.

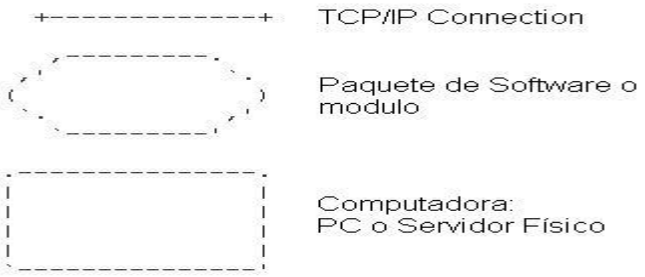


Gráfico 2

Fuente: jabberstudio.org

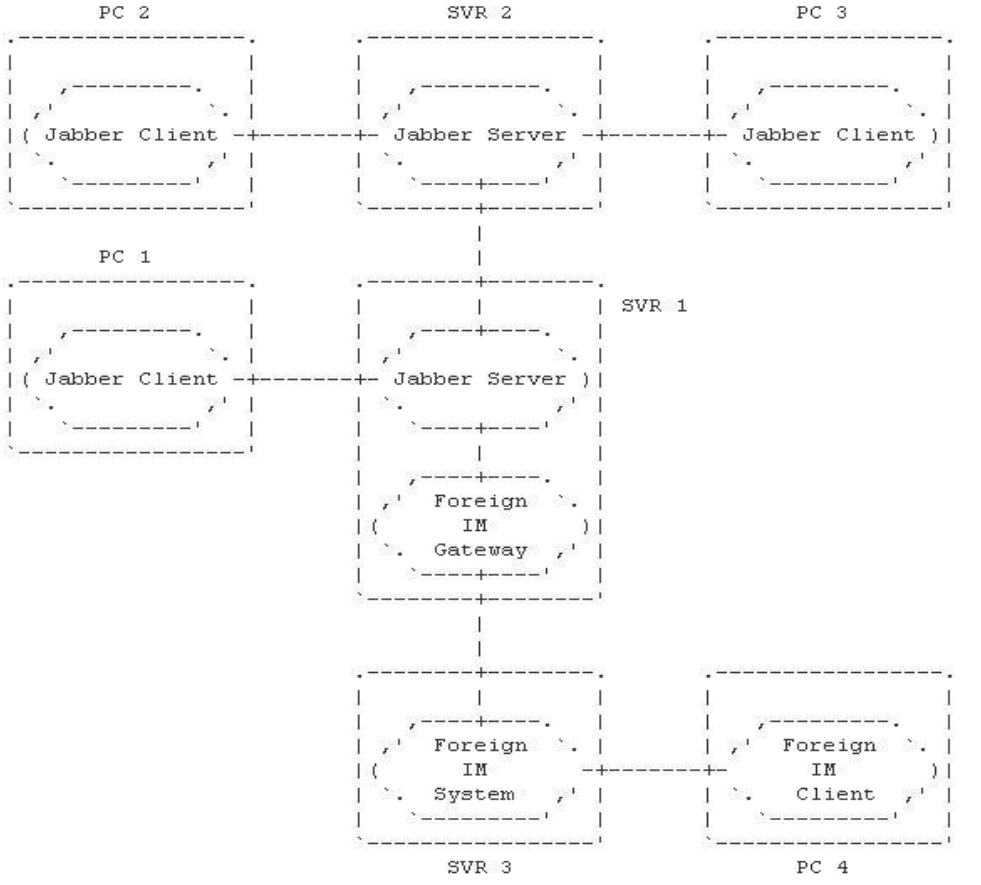


Gráfico 3

Fuente: jabberstudio.org

9.1 Cliente Jabber a Servidor Jabber

En el grafico superior, los clientes de Jabber en *PC 2* y la *PC 3* pueden comunicarse vía Jabber IM proporcionado por *SVR 2*. Ambos clientes tienen cuentas en este servidor, y esta parte del diagrama representaría un sistema *cerrado* o *privado de* Jabber.

Note que un *cliente de Jabber* no es necesariamente un cliente usuario-básico de mensajería instantánea. Por ejemplo, el cliente que funcionaba en *PC 3* pudo realmente ser parte de un Web Server. De este modo, un usuario en *PC 1* pudo hacer actualizaciones al Web Server usando a un cliente de Jabber.

9.2 Servidor Jabber a Servidor Jabber

La comunicación entre *SVR 1* y *SVR 2* demuestra cómo Jabber emplea una arquitectura distribuida. Los clientes en *PC 1* y *PC 2* pueden comunicarse con uno, aun cuando estos clientes tienen cuentas en los servidores separados. Cada cliente necesita saber solamente la dirección del cliente con quien desea comunicarse.

Los servidores de Jabber que funcionan en *SVR 1* y *SVR 2* confían en el servicio del Domain Name (DNS) para las operaciones de búsqueda de la

dirección para comunicarse con uno. El sistema de mensajería instantánea de Jabber se asemeja a la arquitectura de red del correo electrónico proporcionada por POP y el smtp, el email más ampliamente utilizado en el Internet. Jabber no confía en una granja centralizada del servidor. Así, Jabber es fácilmente escalable, y puede ser utilizado como sistema cerrado o abierto.

9.3 Servidor Jabber a otra IM.

En el grafico 1.3 *SVR 1* se comunica con una *entrada extranjera IM* que funciona en la misma máquina. Esta entrada puede comunicarse con un *sistema extranjero IM*, tal como AOL, MSN, Yahoo o el IRC. La conexión con esta entrada de la comunicación permite que los clientes en *PC 1* y *PC 4* se comuniquen a pesar de que el usuario en *PC 1* está funcionando software del cliente de Jabber, mientras que el usuario en *PC 4* es software corriente.

El ejemplo *extranjero IM* demuestra la flexibilidad que Jabber proporciona. Los protocolos de XMPP dictan el sistema de estándares XML-básicos de la comunicación.

10. Instalación del Cliente

Para instalar el cliente CSJabber se requiere el JDK, se recomienda la versión 1.5.0 o superior. Tanto para Windows como para Linux.

Con ello evitamos tener que incluir el paquete jso-ful.jar haciendo uso de CLASSPATH.

Se debe guardar todos los *.class y jso-full.jar en una misma carpeta y realizar las acciones que siguen sobre este directorio.

10.1 Instalación bajo Sistema Operativo Linux

El primer paso será descomprimir esta librería dentro de la carpeta del proyecto.

Luego procedemos a teclear el siguiente comando (desde el Terminal de Linux), con ello logramos compilar todas las clases.

```
PATH=/usr/javaa/jdk1.5.0_06/bin
javac *. java
```

A continuación tecleamos:

```
Java Main
```

Y así logramos ejecutar el programa.

Se recomienda compilar los programas fuentes por el administrador y dejar solo los ejecutables en las máquinas del cliente, para que los usuarios puedan realizar solamente el último pasó para correr el CSJabber.

10.2 Instalación bajo Sistema Operativo Windows XP

Primero debemos entrar al directorio donde se encuentran las clases desde la ventana de comandos de Windows.

Luego debemos descomprimir jso-full.jar en el directorio donde se encuentran almacenadas las clases.

Lo siguiente es ejecutar el comando:

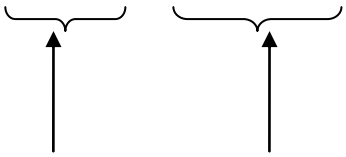
```
Java Main
```

Luego nos aparecerá la pantalla de inicio de CSJabber.

11. Estándar de Desarrollo del Proyecto

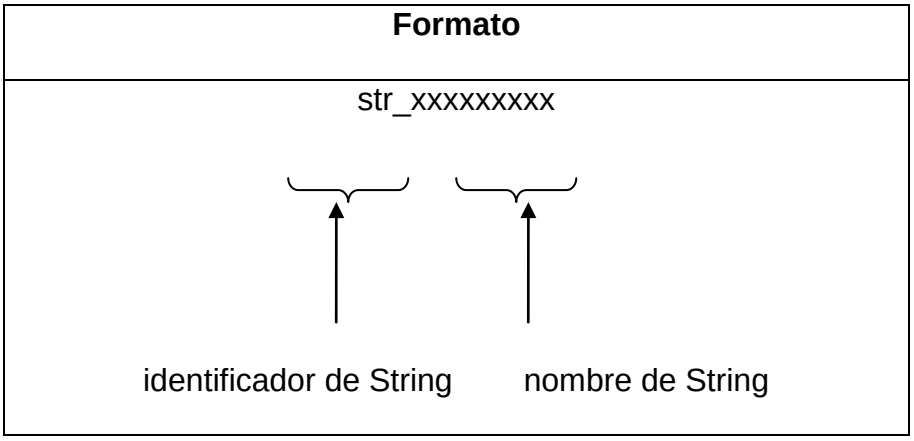
Para una mejor comprensión de la programación dentro de los módulos se ha creado una serie de estándares para reconocer más fácilmente los tipos de objetos y datos que se han utilizados así tenemos la siguiente descripción:

11.1 Identificador de tipo de variables:

Formato
xxx_xxxxxxxxx  identificador tipo de variables nombre del variable

Tipo de variable	Abreviatura	Ejemplo
Bolean	bo	bo_continue
Char	chr	chr_variables
Byte	byt	byt_variables
Short	sht	sht_variable
Int	int	Int_port
Long	lng	lng_inicio
Flota	flt	flt_variable
Double	dou	Dou_variable
Casos especiales		
BigInteger	bint	Bint_ks
BigDecimal	bdec	Bdec_variable

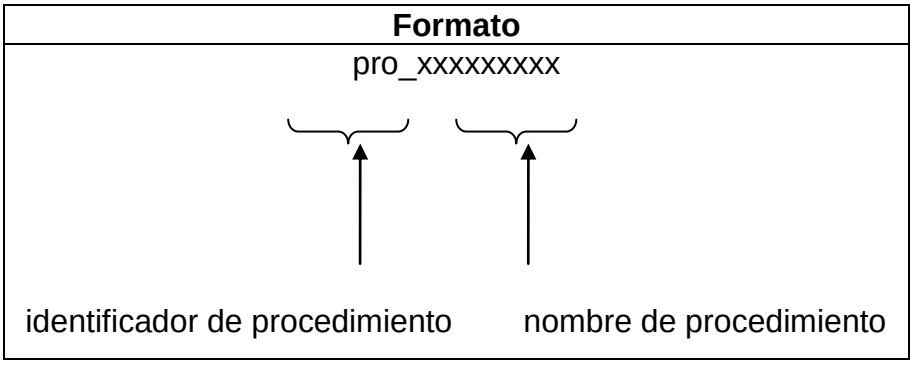
11.2 Clase de String y Wrappers(envolturas)



Ejemplo: Str_Password;

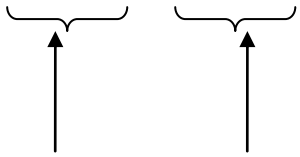
Las clases de java pueden ser: String y StringBuffer y los wrappers. Existe una clase Wrapper para cada uno de los tipos primitivos numéricos, esto es, existen las clases Byte, Short, Integer, Long, Float y Double (obsérvese que los nombres empiezan por mayúscula, siguiendo la nomenclatura típica de ***Java***).

11.3 Nombres para procedimientos



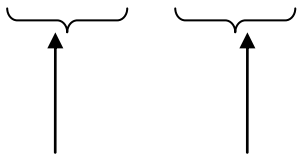
Ejemplo: pro_desconectar

11.4 Nombres para funciones

Formato
ftn_xxxxxxxxx  identificador de la función nombre de la función

Ejemplo: ftn_generarClau(128);

11.5 Nombres de las clases

Formato
cla_xxxxxxxxx  Identificador de clase nombre de clase

11.6 Componentes visuales (Swing)

Control	Prefijo	Ejemplo
JTextPane	jtp	Jtp_lista
JTextField	txt	txtmensaje
JButton	btn	btn_enviar
JScrollPanel	jsp	Jsp
JChekBox	Chk	Chkcrip
JLabel	Lbl	lbl_host
JPanel	Jpl	Jpl_panel

12. Descripción de la Bases de Datos

La base de datos del proyecto es auto generada al momento de la instalación del servidor su diagrama es el siguiente:

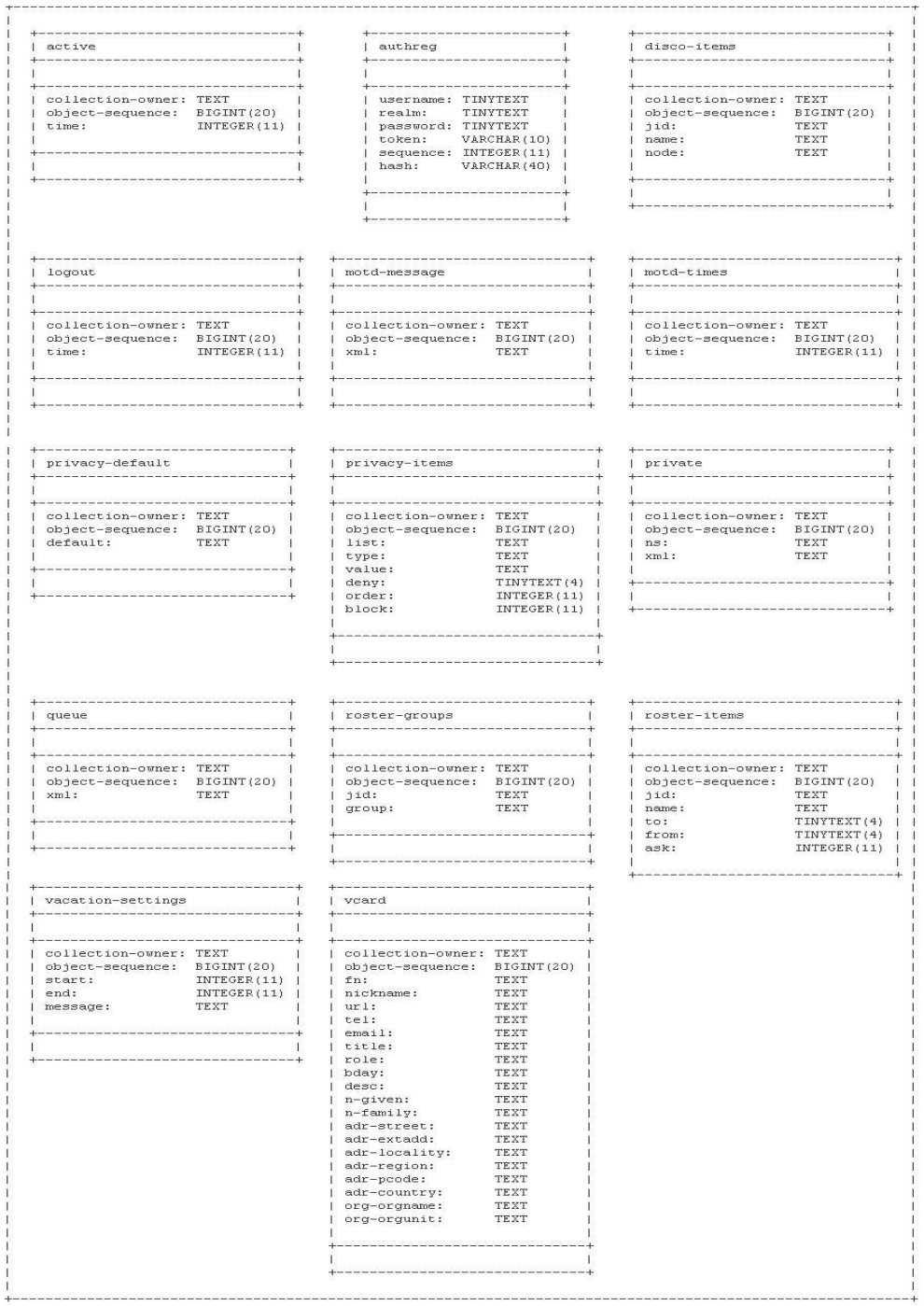


Gráfico 4

Fuente: jabber.org

13. Diccionario de la Base de Datos

A continuación se describe la base de datos Jabberd2 y cada tabla con sus campos: (14 tablas)

Nombre de la tabla :	Active				
Alias:	Ninguno				
Descripción del contenido:	Almacena la fecha/hora sobre las cuales cada cuenta primero llegó a ser activa.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**extra**
Collection-owner	Text		Mul		
Object-sequence	Bigint(20)		Pri	Null	Auto_increment
Time	Int(11)	Yes		Null	

Nombre de la tabla :	Authreg				
Alias:	Ninguno				
Descripción del contenido:	Contiene la información de la autentificación, incluyendo el username, el realm y la contraseña. La tabla del authreg maneja el aspecto de la autentificación para Jabberd cuando se usa MyQSL				
Nombre de Campo	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Username	Text	Yes	Mul	Null	
Realm	Tinytext	Yes	Mul	Null	
Password	Tinytext	Yes		Null	
Token	Varchar(10)	Yes		Null	
Sequence	Int(11)	Yes		null	
Hash	Varchar(40)	Yes		Null	

Nombre de la tabla :	disco-items				
Alias:	Ninguno				
Descripción del contenido:	Almacena la información persistente del descubrimiento de modo que esté disponible para la recuperación fuera de línea.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text				
Object-sequence	Bigint(20)				
Jid	Text				
Name	Text				
Node	Text				

Nombre de la tabla :	logout				
Alias:	Ninguno				
Descripción del contenido:	Los almacenes JID y el timestamp para la mayoría del usuario reciente registran hacia fuera.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text		Mul		
Object-sequence	Bigint(20)		pri	null	Auto-increment
Time	*Int(11)*	*Yes*		*Null*	

Nombre de la tabla :	motd-message				
Alias:	Ninguno				
Descripción del contenido:	Almacena el mensaje del día (MOTD) en formato de XML.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text				
Object-sequence	Bigint(20)				
Xml	Text				

Nombre de la tabla :	motd-times				
Alias:	Ninguno				
Descripción del contenido:	Expedientes JID y timestamps para el recibo de MOTD.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text				
Object-sequence	Bigint(20)				
Time	Integer(11)				

Nombre de la tabla :	privacy-default				
Alias:	Ninguno				
Descripción del contenido:	Almacena el nombre de la lista actual funcionando para un usuario así puede ser hecha activa en el momento del arranque.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**extra**
Colecction-owner	text				
Object-sequence	Bigint(20)				
Default	text				

Nombre de la tabla :	privacy-items				
Alias:	Ninguno				
Descripción del contenido:	Listas del aislamiento del usuario de los almacenes (listas negras/whitelists).				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**extra**
Collection-owner	Text				
Object-sequence	Bigint(20)				
List	Text				
Type	Text				
Value	Text				
Deny	Tinytext(4)				
Order	Integer(11)				
Block	Integer(11)				

Nombre de la tabla:	Private				
Alias:	Ninguno				
Descripción del contenido:	Proporciona el almacenaje privado de XML para las aplicaciones tales como preferencias o bookmarks del usuario.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text		Mul		
Object-sequence	Bigint(20)		Pri	Null	Auto_increment
Ns	Text	Yes		Null	
Xml	Mediumtext	Yes		Null	

Nombre de la tabla :	Queue				
Alias:	Ninguno				
Descripción del contenido:	Los almacenes hicieron cola mensajes en formato de XML.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text		Mul		
Object-sequence	Bigint(20)		Pri	Null	Auto-increment
Xml	Mediumtext	Yes		null	

Nombre de la tabla :	Roster-groups				
Alias:	Ninguno				
Descripción del contenido:	Almacena artículos de la lista del usuario solamente para esos artículos de la lista que tengan un grupo asignado.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text				
Object-sequence	Bigint(20)				
Jid	Text				
Group	Text				

Nombre de la tabla :	roster-items				
Alias:	Ninguno				
Descripción del contenido:	Almacena artículos de la lista del usuario, incluyendo estado de la autorización***.***				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text				
Object-sequence	Bigint(20)				
Jid	Text				
Name	Text				
To	Tinytext(4)				
From	Tinytext(4)				
Ask	Integer(11)				

Nombre de la tabla :	vacation-settings				
Alias:	Ninguno				
Descripción del contenido:	Maneja ajustes de las vacaciones, incluyendo comienzo, extremo y mensaje.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text				
Object-sequence	Bigint(20)				
Start	Integer(11)				
End	Integer(11)				
Message	Text				

Nombre de la tabla :	Vcard				
Alias:	Ninguno				
Descripción del contenido:	Almacena la información de la tarjeta del usuario.				
Campos:	**Tipo**	**Null**	**Key**	**Default**	**Extra**
Collection-owner	Text		Mul		
Object-sequence	Bigint(20)		Pri	Null	Auto-increment
Fn	Text	Yes		Null	
Nickname	Text	Yes		Null	
url	Text	Yes		Null	
Tel	Text	Yes		Null	
Email	Text	Yes		Null	
Title	Text	Yes		Null	
Role	Text	Yes		Null	
bday	Text	Yes		Null	
Desc	Text	Yes		Null	
n-given	Text	Yes		Null	
n-family	Text	Yes		Null	
Adr-street	Text	Yes		Null	
Adr-extadd	Text	Yes		Null	
Adr-locality	Text	Yes		Null	
Adr-region	Text	Yes		Null	
Adr-pcode	Text	Yes		Null	
Adr-country	Text	Yes		Null	
Org-orgname	Text	Yes		Null	
Org-orgunit	Text	Yes		Null	

MANUAL DE USUARIO

Manual de Usuario de un Cliente Seguro de Mensajería Instantánea Basado en el Protocolo JABBER.

El Sistema CSJabber es un cliente Seguro de mensajería instantánea basada en Jabber que es un protocolo de comunicación gratuito, abierto, público y comprensible. Este sistema permite a sus usuarios conectarse a el servidor, obtener la lista de contactos, abrir una conversación con otros clientes, enviar archivos y encriptar los mensajes.

1. Propósito

Este manual ha sido realizado con la finalidad de que se convierta en una guía Fácil al momento de utilizar el Sistema Cliente Seguro Jabber (CSJabber.)

2. Aplicabilidad

El funcionamiento de nuestra aplicación es fácilmente comprensible y está estructurado de la siguiente manera.

3 PRINCIPIOS DE INTERFAZ DE USUARIO

3.1 Elementos de la Ventana Principal

La interacción entre el usuario y el sistema se realiza a través de las diferentes opciones del menú que podrán ser usadas por cualquier Usuario. Como podemos observar en la siguiente pantalla.

1. Menú de Archivo uno de los principales para el uso del Sistema CSJabber; a través de esta opción podemos interactuar con la mayor parte del Sistema.

2. Menú de Contacto que muestra la lista de contactos que posee el usuario y de esa manera puede iniciar una sesión de Chat con la persona que se encuentre en línea.

3. Menú de Ayuda; nos indica el manejo de cada una de las ventanas de CSJabber.

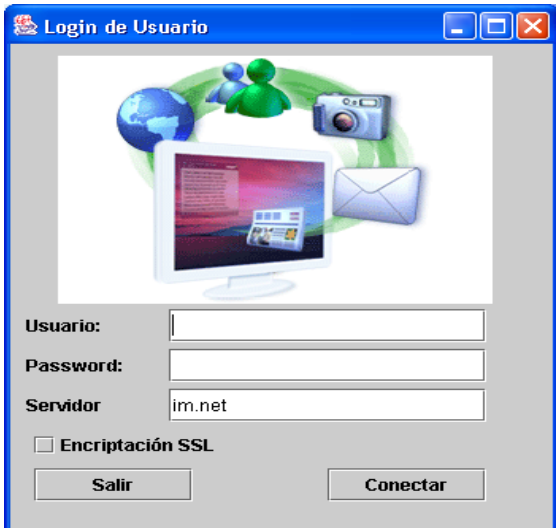
3.2 Menú Archivo.

Para ingresar al sistema CSJabber se debe hacer clic sobre la opción del menú Archivo donde se despliega la siguiente pantalla.

3.2.1 Inicio de Sesión.

Al Iniciar sesión el sistema solicitará los datos de identificación del usuario, su password y el servidor del Sistema, estos datos son imprescindible para iniciar la conexión con servidor y de esta forma poder iniciar la conversación con otros usuarios.

Además el Jcheckbox Encriptación permite al usuario enviar sus datos encriptados y autentificarse. Esta pantalla también, posee los botones Salir y Conectar.

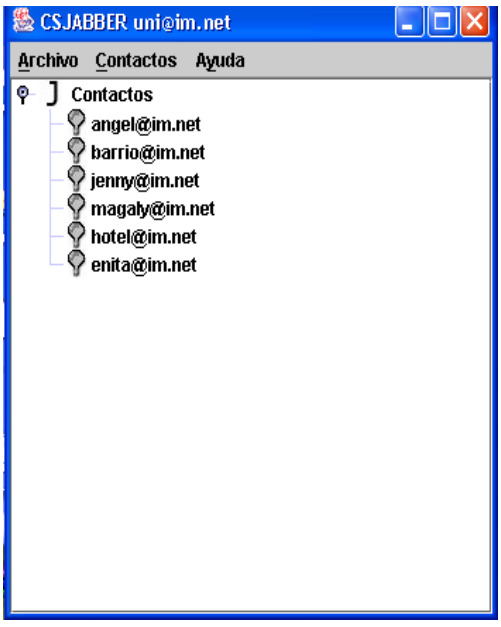


En usuario se debe ingresar su cuenta la misma, que tendrá el siguiente formato ***nombre_usuario@servidor*** en este caso el servidor es **im.net**

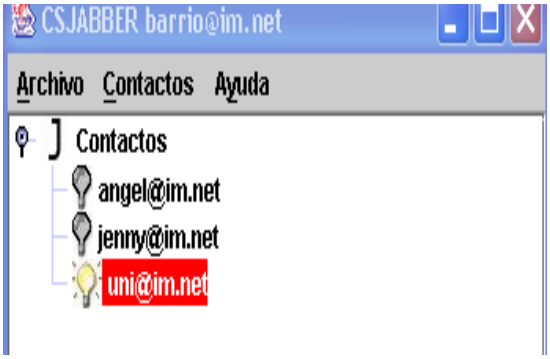
Luego deberá ingresar el ***password*** es decir su clave, y en servidor deberá ingresar el nombre del servidor en el que este su cuenta.

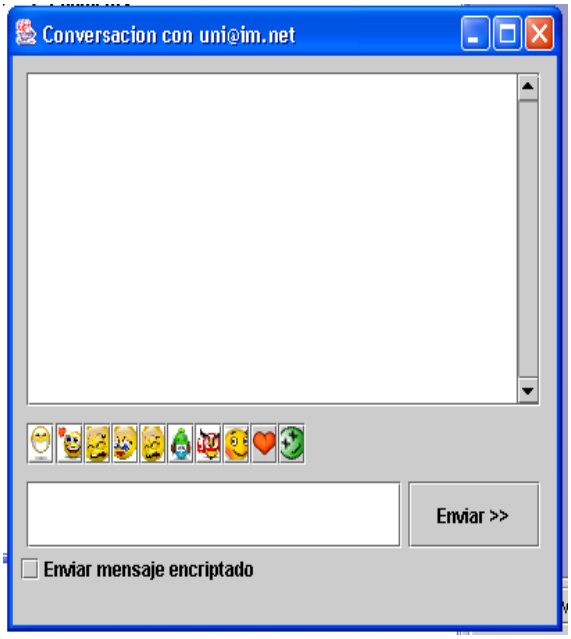
El jcheckbox de Encriptación es parte de la seguridad de la información y se lo ejecuta con tan solo dar click en su casillero, este proceso se realiza de manera interna; una vez ingresado los datos de clic en el botón ***conectar***.

Si los datos ingresados son correctos se muestran todos los contactos que el usuario posee y además el estado de cada uno de ellos en el momento de la conexión.



Para iniciar la conversación con un usuario determinado simplemente de clic sobre dicho usuario y se mostrara la ventana de chatear. Los iconos brillantes muestran los usuarios conectados.





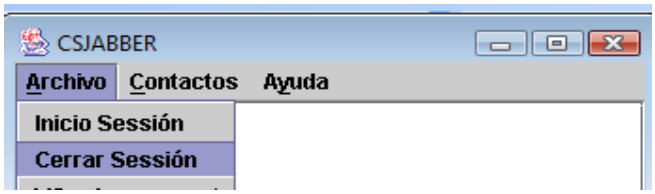
Por tanto podrá iniciar la conversación en el Chat individualmente con cada uno de los usuarios seleccionado. Los usuarios podrán enviarse emoticones o smileys, además podrán hacer trasferencia de archivo.

Al dar clic en ***Enviar.***

En la parte inferior de la pantalla se encuentra un casilla que permite enviar el mensaje de forma cifrada y que solo su destinatario pueda leerlo, para esto deberá seleccionar el casillero de Encriptación.

3.2.2 Cerrar Sesión

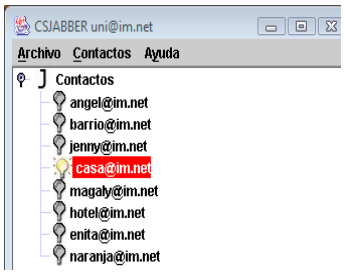
La opción ***Cerrar sesión*** le permite al usuario desconectarse del servidor y por lo tanto terminar la conversación.



3.2.3 Transferir Archivo.

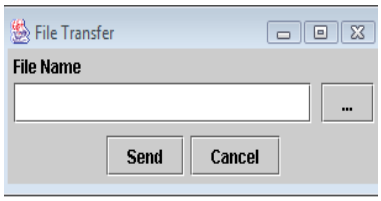
Para enviar un archivo se debe:

1. Seleccione de la pantalla principal el usuario destinatario.

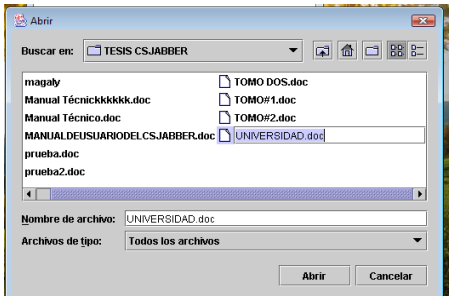


2. De clic en menú Archivo y luego seleccione Transferir Archivo.

3. Al desplegarse esta pantalla seleccione el botón el mismo que le permitirá buscar el archivo a enviar.



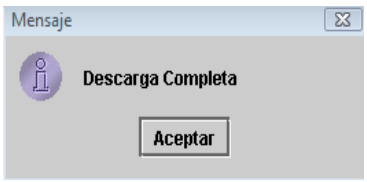
4. Una vez seleccionado el documento a enviar de clic en botón abrir.



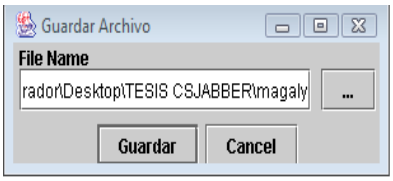
5. La ruta del archivo seleccionado se agregara en la pantalla File Ttansfer, para enviarlo de Clic en el boton *Send*

Para recibir archivo se debe:

1. Cuando un archivo es transferido el usuario destinatario visualiza la siguiente pantalla, para descargar el archivo al PC debe dar clic en el botón ***Aceptar.***

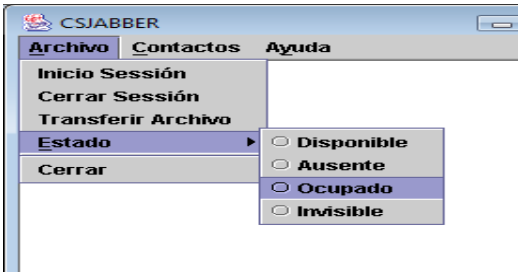


2. Luego de clic en el botón ... y seleccione el directorio donde se guardará el archivo recibido, una vez seleccionado de clic en el botón Guardar.



3.2.4 Estado

Para cambiar el estado el usuario deber ir al menú Archivo y seleccionar ***Estado***, donde deberá escoger una de las opciones disponible.



Las cuatro opciones que se visualizan en ***Estado*** permiten realizar lo siguiente:

- Disponible : Resalta si estoy en línea
- Ausente: Indica que el usuario no esta disponible para el Chat.
- Ocupado : Indica que esta realizando otra tarea y por un momento desea estar fuera del Chat

- Invisible: Si no deseo seguir chateando con ningún usuario. No aparecerá en la lista de los otros usuarios de CSJabber.

3.2.5 Cerrar

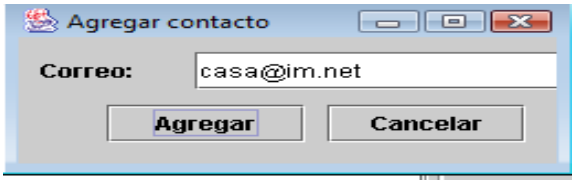
La opción **CERRAR** deshabilita las opciones del usuario desconectándolo del servidor.

3.3 Menú Contacto

Este menú tiene dos opciones

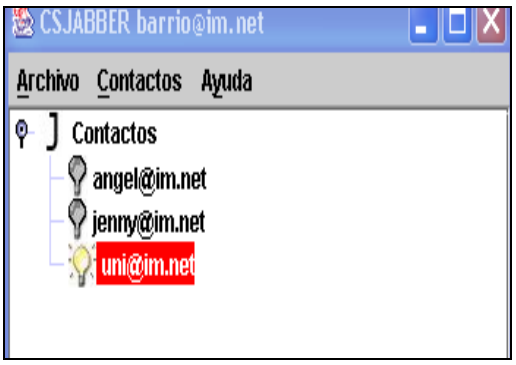
3.3.1 Agregar un contacto

Escriba la cuenta del usuario a agregar (nombre_usuario@servidor), luego de clic en agregar.



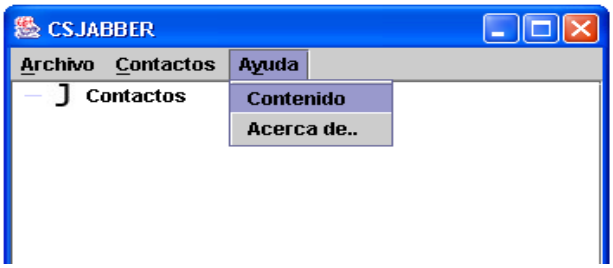
3.3.2 Eliminar un contacto

Se debe seleccionar el usuario a borrar de la lista de contactos y de clic en menú ***Contactos*** y después de clic en ***Eliminar un contacto***.



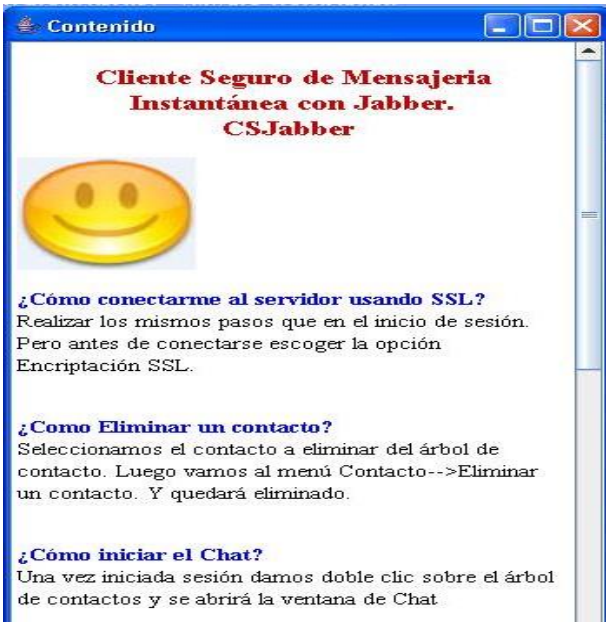
3.4 Ayuda

La Opción **Ayuda** permite al usuario conocer la funcionalidad del programa de mensajería CSJabber y esta compuesto por dos opciones.



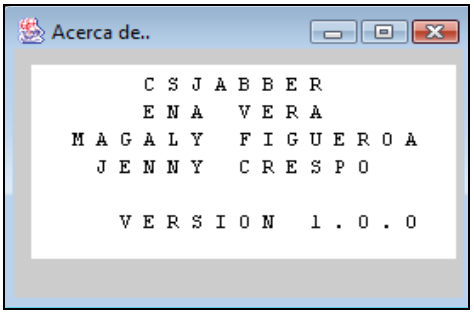
3.4.1 Contenido.

Nos mostrara un glosario de preguntas que normalmente se hacen los usuarios sobre como funciona el sistema.


Contenido
Cliente Seguro de Mensajeria Instantánea con Jabber.
CSJabber
¿Cómo conectarme al servidor usando SSL?
Realizar los mismos pasos que en el inicio de sesión. Pero antes de conectarse escoger la opción Encriptación SSL.
¿Como Eliminar un contacto?
Seleccionamos el contacto a eliminar del árbol de contacto. Luego vamos al menú Contacto-->Eliminar un contacto. Y quedará eliminado.
¿Cómo iniciar el Chat?
Una vez iniciada sesión damos doble clic sobre el árbol de contactos y se abrirá la ventana de Chat

3.4.2 Acerca del Sistema

Presenta el nombre del sistema, la versión del Software y el nombre de los creadores.



RECOMENDACIONES

a. Vea el sitio: http://www.gnu.org/software/libidn/&prev=/search%3Fq%3Djabberd2%26hl%3Des%26sa%3DX para obtener mayor información sobre libidn y como instalar.

b. Considere que Jabberd requiere más que una instalación mínima de MySQL. Además de la instalación básica de MySQL, Jabberd requiere development libraries y headers estén instalados. Realizar una instalación máxima, o instalar el servidor, los programas del cliente, las bibliotecas y los archivos de header, y las bibliotecas dinámicas del cliente por separado. Puede ser necesario desinstalar tu instalación actual de MySQL para instalar las bibliotecas adicionales.